EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
S1	41	alexander near keller.in.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:20
S2	55	joseph near hellerstein.in.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:26
S3	3	vijaya near krishnan.in.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:27
S4	38	joel near wolf.in.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:29
S5	33	kun-lung near wu.in.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:31
S6	33	kun near lung near wu.in.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:31
S7	66558	(ibm or "international business machines").as.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:32
S8	1069	S7 and (chang$4 and order).clm.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:35
S9	2	S7 and (chang$4 and order$4 and task and relationship).clm.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:36
S10	37	S7 and (chang$4 and order$4 and task$4).clm.	US-PGPUB; USPAT	OR	ON	2006/11/13 10:36
S11	20	("4751635"|"5493682"|"5721824"|"5805891"|"5835777"|"5867714"|"5953533"|"5960196"|"5999740"|"6009525").PN.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 10:41
S12	4	("5890166" "6192368").pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 11:01
S13	1482	717/174-178.ccls.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 12:30
S14	96	S13 and ((order$4 or sequenc$4) and task and constraint)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 12:32

EAST Search History

S15	93729	chang$4 near (order$4 or sequenc$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 13:27
S16	3000	order$4 near (constraint or restriction)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 13:27
S17	284	S15 and S16	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 13:28
S18	277	S17 and (task or step)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 13:29
S19	90	S18 and install$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 13:29
S20	84	S19 and (@pd<"20040227" or @ad<"20040227" or @prad<"20040227" or @rlad<"20040227")	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 13:30
S21	49	("5359730" | "5493682" | "5504905" | "5634058" | "5649187" | "5666501").PN. OR ("5721824").URPN.	US-PGPUB; USPAT; USOCR	OR	ON	2006/11/13 13:40

EAST Search History

S22	101	("4833638" "5544301" "5615109" "20050192979" "5533179" "5740236" "5465362" "5465363" "5788504" "5889530" "5973702" "4389706" "4400694" "4475156" "4809170" "4827400" "4965500" "5233510" "5235700" "5339389" "5359730" "5371883" "5386464" "5485615" "5515491" "5522025" "5524199" "5524200" "5533116" "5549118" "5555368" "5581664" "5598537" "5610839" "5615326" "5619409" "5619638" "5634056" "5668997" "5727203" "5737559" "5745880" "5758062" "5764754" "5768510" "5796969" "5812849" "5835918" "5835735" "5848246").pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 14:18
S23	231	S13 and (dependency and (order$4 or sequenc$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 15:13
S24	4542	task near (graph or chart or list)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 16:05
S25	985	S24 and ((order$4 or sequenc$4) with chang$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 16:05
S26	211	S25 and dependency	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 16:10
S27	11	(US-20040243996-$ or US-20040243995-$).did. or (US-5721824-$ or US-5835777-$ or US-5634056-$ or US-6192368-$ or US-5890166-$ or US-7058942-$ or US-6681391-$ or US-6675382-$ or US-6052707-$).did.	US-PGPUB; USPAT	OR	ON	2006/11/13 16:55

EAST Search History

S28	2	S27 and (time with (estimat$4 or approximat$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/13 16:55
S29	1250058	time with (estimat$4 or approximat$4 or rang$4 or interval)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 07:36
S30	909578	time near5 (estimat$4 or approximat$4 or rang$4 or interval)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 07:36
S31	12863	S30 and ((order$4 or sequenc$4) with (task or job))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 07:37
S32	32	S31 and (component adj dependency)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 08:43
S33	255	S31 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 08:43
S34	19	("5748959").URPN.	USPAT	OR	ON	2006/11/14 09:41
S35	2090	S31 and ((parallel or concurrently or simulatenously) near (process$4 or execut$4))	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 09:44
S36	98	S35 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 09:44

EAST Search History

S37	15	(US-20040243996-$ or US-20040243995-$).did. or (US-5721824-$ or US-6192368-$ or US-5835777-$ or US-5890166-$ or US-7058942-$ or US-5634056-$ or US-6681391-$ or US-6675382-$ or US-6052707-$ or US-5748959-$ or US-6230313-$ or US-6952825-$ or US-6973417-$).did.	US-PGPUB; USPAT	OR	ON	2006/11/14 10:23
S39	6	S37 and (network and name and identifier)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:24
S40	15	("5721824" | "5758154" | "5761380" | "5870611" | "5999181" | "6119122" | "6158001" | "6286010" | "6381742" | "6442754" | "6473771" | "6487713" | "6556223" | "6560776" | "6725452").PN.	US-PGPUB; USPAT; USOCR	OR	ON	2006/11/14 10:28
S41	69	(unique adj physical adj (identifier or id)) and network	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:37
S42	48879	(unique near3 (identifier or id)) and network	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:38
S43	354	S42 and "logical name"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:38
S44	16	S43 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:38

EAST Search History

S45	361	(target adj system) and (unique near3 (id or identification)) and name	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:40
S46	35	S45 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:44
S47	46727	network and (global or unique) near (identification or id or number or address)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:49
S48	45924	S47 and (client or system)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:46
S49	770	S48 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:48
S50	9	S49 and (logical adj name)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:47
S51	75079	(global or unique) near (identification or id or number or address)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:52
S52	253	S51 and "logical name"	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:50

EAST Search History

S53	9	S52 and "717".clas.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:50
S54	165932	(global or unique or physical) near3 (identification or id or number or address)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:53
S55	9216	S54 and (client adj (computer or system)) and network	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:54
S56	482	S55 and (computer or logical) near name	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 10:55
S57	3	"6918112".pn.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2006/11/14 16:04

Terms used **installation dependency task graph** Found **45,732** of **192,172**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Backtracking intrusions
Samuel T. King, Peter M. Chen
February 2005 **ACM Transactions on Computer Systems (TOCS)**, Volume 23 Issue 1
Publisher: ACM Press
Full text available: pdf(647.38 KB) Additional Information: full citation, abstract, references, index terms

Analyzing intrusions today is an arduous, largely manual task because system administrators lack the information and tools needed to understand easily the sequence of steps that occurred in an attack. The goal of BackTracker is to identify automatically potential sequences of steps that occurred in an intrusion. Starting with a single detection point (e.g., a suspicious file), BackTracker identifies files and processes that could have affected that detection point and displays chains of events i ...

Keywords: Computer forensics, information flow, intrusion analysis

2 Making operating systems more robust: Backtracking intrusions
Samuel T. King, Peter M. Chen
October 2003 **Proceedings of the nineteenth ACM symposium on Operating systems principles**
Publisher: ACM Press
Full text available: pdf(185.10 KB) Additional Information: full citation, abstract, references, citings, index terms

Analyzing intrusions today is an arduous, largely manual task because system administrators lack the information and tools needed to understand easily the sequence of steps that occurred in an attack. The goal of BackTracker is to identify automatically potential sequences of steps that occurred in an intrusion. Starting with a single detection point (e.g., a suspicious file), BackTracker identifies files and processes that could have affected that detection point and displays chains of events i ...

Keywords: computer forensics, information flow, intrusion analysis

3 A multi-level parallelism architecture
Theo Ungerer, Eberhard Zehendner
July 1991 **ACM SIGARCH Computer Architecture News**, Volume 19 Issue 4
Publisher: ACM Press
Full text available: pdf(913.64 KB) Additional Information: full citation, abstract, index terms

This paper describes a parallel programming language and a multi-level parallelism computer architecture, which have been developed by an integrated design process.The

ASTOR language is an imperative intermediate language that combines parallel control constructs with a concise module concept. The ASTOR architecture is directed towards reliable execution of programs written in the ASTOR language, utilizing five levels of parallelism expressed by the language constructs. Structure preservation ha ...

4 High speed on-line backup when using logical log operations

David B. Lomet

May 2000 **ACM SIGMOD Record , Proceedings of the 2000 ACM SIGMOD international conference on Management of data SIGMOD '00**, Volume 29 Issue 2

Publisher: ACM Press

Full text available: pdf(220.69 KB) Additional Information: full citation, abstract, references, citings, index terms

Media recovery protects a database from failures of the stable medium by maintaining an extra copy of the database, called the backup, and a media recovery log. When a failure occurs, the database is "restored" from the backup, and the media recovery log is used to roll forward the database to the desired time, usually the current time. Backup must be both fast and "on-line", i.e. concurrent with on-going update activity. Conventional online backup sequentially copies ...

5 Parsing: Minimal change and bounded incremental parsing

Mats Wirén

August 1994 **Proceedings of the 15th conference on Computational linguistics - Volume 1**

Publisher: Association for Computational Linguistics

Full text available: pdf(608.37 KB) Additional Information: full citation, abstract, references

Ideally, the time that an incremental algorithm uses to process a change should be a function of the size of the change rather than, say, the size of the entire current input. Based on a formalization of "the set of things changed" by an incremental modification, this paper investigates how and to what extent it is possible to give such a guarantee for a chart-based parsing framework and discusses the general utility of a minimality notion in incremental processing.

6 System level modelling: Heterogeneous behavioral hierarchy for system level designs

Hiren D. Patel, Sandeep K. Shukla, Reinaldo A. Bergamaschi

March 2006 **Proceedings of the conference on Design, automation and test in Europe: Proceedings DATE '06**

Publisher: European Design and Automation Association

Full text available: pdf(223.30 KB) Additional Information: full citation, abstract, references

Enhancing productivity for designing complex embedded systems requires system level design methodology and language support for capturing complex design in high level models. For an effective methodology, efficiency of simulation and a sound refinement based implementation path are also necessary. Although some of the recent system level design languages for system level abstractions, several essential ingredients are missing from these. We consider (i) explicit support for multiple models of co ...

7 Fast detection of communication patterns in distributed executions

Thomas Kunz, Michiel F. H. Seuren

November 1997 **Proceedings of the 1997 conference of the Centre for Advanced Studies on Collaborative research**

Publisher: IBM Press

Full text available: pdf(4.21 MB) Additional Information: full citation, abstract, references, index terms

Understanding distributed applications is a tedious and difficult task. Visualizations based on process-time diagrams are often used to obtain a better understanding of the execution of the application. The visualization tool we use is Poet, an event tracer developed at the University of Waterloo. However, these diagrams are often very complex and do not provide the user with the desired overview of the application. In our

experience, such tools display repeated occurrences of non-trivial commun ...

8 Bridging the gap between technical and social dependencies with Ariadne

Erik Trainer, Stephen Quirk, Cleidson de Souza, David Redmiles
October 2005 **Proceedings of the 2005 OOPSLA workshop on Eclipse technology eXchange eclipse '05**
Publisher: ACM Press
Full text available: pdf(312.35 KB) Additional Information: full citation, abstract, references, index terms

One of the reasons why large-scale software development is difficult is the number of dependencies that software engineers need to face; e.g., dependencies among the software components and among the development tasks. These dependencies create a need for communication and coordination that requires continuous effort by software developers. Empirical studies, including our own, suggest that technical dependencies among software components create social dependencies among the software developers ...

Keywords: collaborative software development, program dependencies, social dependencies

9 Visualization of parallel and distributed systems: Adding parallelism to visual data flow programs

Philip Cox, Simon Gauvin, Andrew Rau-Chaplin
May 2005 **Proceedings of the 2005 ACM symposium on Software visualization**
Publisher: ACM Press
Full text available: pdf(278.29 KB) Additional Information: full citation, abstract, references

Programming in parallel is an error-prone and complex task compounded by the lack of tool support for both programming and debugging. Recent advances in compiler-directed shared memory APIs, such as OpenMP, have made shared-memory parallelism more widely accessible for users of traditional procedural languages: however, the mechanisms provided are difficult to use and error-prone. This paper examines the use of visual notations for data flow programming to enable the creation of shared memory pa ...

Keywords: data flow, parallel, visual language

10 Parallel execution of prolog programs: a survey

Gopal Gupta, Enrico Pontelli, Khayri A.M. Ali, Mats Carlsson, Manuel V. Hermenegildo
July 2001 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 23 Issue 4
Publisher: ACM Press
Full text available: pdf(1.95 MB) Additional Information: full citation, abstract, references, citings, index terms

Since the early days of logic programming, researchers in the field realized the potential for exploitation of parallelism present in the execution of logic programs. Their high-level nature, the presence of nondeterminism, and their referential transparency, among other characteristics, make logic programs interesting candidates for obtaining speedups through parallel execution. At the same time, the fact that the typical applications of logic programming frequently involve irregular computatio ...

Keywords: Automatic parallelization, constraint programming, logic programming, parallelism, prolog

11 Making snapshot isolation serializable

Alan Fekete, Dimitrios Liarokapis, Elizabeth O'Neil, Patrick O'Neil, Dennis Shasha
June 2005 **ACM Transactions on Database Systems (TODS)**, Volume 30 Issue 2

Publisher: ACM Press
Full text available: pdf(722.95 KB) Additional Information: full citation, abstract, references, index terms

Snapshot Isolation (SI) is a multiversion concurrency control algorithm, first described in Berenson et al. [1995]. SI is attractive because it provides an isolation level that avoids many of the common concurrency anomalies, and has been implemented by Oracle and Microsoft SQL Server (with certain minor variations). SI does not guarantee serializability in all cases, but the TPC-C benchmark application [TPC-C], for example, executes under SI without serialization anomalies. All major database s ...

Keywords: Concurrency control, anomaly, consistency, multiversion concurrency, serializability, snapshot isolation, weak isolation

12 Using UML for software process modeling

Dirk Jäger, Ansgar Schleicher, Bernhard Westfechtel
October 1999 **ACM SIGSOFT Software Engineering Notes , Proceedings of the 7th European software engineering conference held jointly with the 7th ACM SIGSOFT international symposium on Foundations of software engineering ESEC/FSE-7**, Volume 24 Issue 6
Publisher: Springer-Verlag, ACM Press
Full text available: pdf(1.44 MB) Additional Information: full citation, references, citings, index terms

Keywords: software engineering tools and environments, software process models

13 Courses: Exploiting perception in high-fidelity virtual environments

Mashhuda Glencross, Alan G. Chalmers, Ming C. Lin, Miguel A. Otaduy, Diego Gutierrez
July 2006 **Material presented at the ACM SIGGRAPH 2006 conference SIGGRAPH '06**
Publisher: ACM Press
Full text available: pdf(5.25 MB) Additional Information: full citation, abstract

This course introduces high-fidelity virtual environments and explains the key components required to build compelling environments. Then it details perceptually inspired techniques that facilitate high-fidelity rendering, collaboration, and complex interaction in these virtual environments. Particular emphasis is placed on real applications, with several live demonstrations.

14 Using temporal hierarchies to efficiently maintain large temporal databases

Thomas Dean
October 1989 **Journal of the ACM (JACM)**, Volume 36 Issue 4
Publisher: ACM Press
Full text available: pdf(2.94 MB) Additional Information: full citation, abstract, references, citings, index terms

Many real-world applications involve the management of large amounts of time-dependent information. Temporal database systems maintain this information in order to support various sorts of inference (e.g., answering questions involving propositions that are true over some intervals and false over others). For any given proposition, there are typically many different occasions on which that proposition becomes true and persists for some length of time. In this paper, these occasions are re ...

15 A formal framework for component deployment

Yu David Liu, Scott F. Smith
October 2006 **ACM SIGPLAN Notices , Proceedings of the 21st annual ACM SIGPLAN conference on Object-oriented programming languages, systems, and applications OOPSLA '06**, Volume 41 Issue 10
Publisher: ACM Press
Full text available: Additional Information:

pdf(592.54 KB) full citation, abstract, references, index terms

Software deployment is a complex process, and industrial-strength frameworks such as .NET, Java, and CORBA all provide explicit support for component deployment. However, these frameworks are not built around fundamental principles as much as they are engineering efforts closely tied to particulars of the respective systems. Here we aim to elucidate the fundamental principles of software deployment, in a platform-independent manner. Issues that need to be addressed include deployment unit design ...

Keywords: component, deployment, version

16 Logical logging to extend recovery to new domains

David Lomet, Mark Tuttle
June 1999 **ACM SIGMOD Record , Proceedings of the 1999 ACM SIGMOD international conference on Management of data SIGMOD '99**, Volume 28 Issue 2
Publisher: ACM Press

Full text available: pdf(1.65 MB) Additional Information: full citation, abstract, references, citings, index terms

Recovery can be extended to new domains at reduced logging cost by exploiting "logical" log operations. During recovery, a logical log operation may read data values from any recoverable object, not solely from values on the log or from the updated object. Hence, we needn't log these values, a substantial saving. In [8], we developed a redo recovery theory that deals with general log operations and proved that the stable database remains recoverable when it is explained in terms ...

17 Continuous program optimization: A case study

Thomas Kistler, Michael Franz
July 2003 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 25 Issue 4
Publisher: ACM Press

Full text available: pdf(877.67 KB) Additional Information: full citation, abstract, references, index terms, review

Much of the software in everyday operation is not making optimal use of the hardware on which it actually runs. Among the reasons for this discrepancy are hardware/software mismatches, modularization overheads introduced by software engineering considerations, and the inability of systems to adapt to users' behaviors.A solution to these problems is to delay code generation until load time. This is the earliest point at which a piece of software can be fine-tuned to the actual capabilities of the ...

Keywords: Dynamic code generation, continuous program optimization, dynamic reoptimization

18 Computing curricula 2001

September 2001 **Journal on Educational Resources in Computing (JERIC)**
Publisher: ACM Press

Full text available: pdf(613.63 KB) html(2.78 KB) Additional Information: full citation, references, citings, index terms

19 Technical reports

SIGACT News Staff
January 1980 **ACM SIGACT News**, Volume 12 Issue 1
Publisher: ACM Press

Full text available: pdf(5.28 MB) Additional Information: full citation

20 Creating minimal vertex series parallel graphs from directed acyclic graphs
Margaret Mitchell
January 2004 **Proceedings of the 2004 Australasian symposium on Information Visualisation - Volume 35 APVis '04**
Publisher: Australian Computer Society, Inc.
Full text available: pdf(134.95 KB) Additional Information: full citation, abstract, references, index terms

Visualizing information whose underlying graph is directed acyclic is easier if the graph is Minimal Vertex Series Parallel (MVSP). We present method of transforming any directed acyclic graph into one that is MVSP. This enables an easy visualization of the information contained in the original graph. We illustrate this by an example of a schedule that contains parallelism.

Keywords: directed acyclic graph, graph decomposition, minimal vertex series parallel, parallelism, preservation of partial order

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next